Министерство науки и высшего образования Российской Федерации Федеральное государственное автономное учреждение высшего образования «Пермский национальный исследовательский политехнический университет»

ПНИПУ

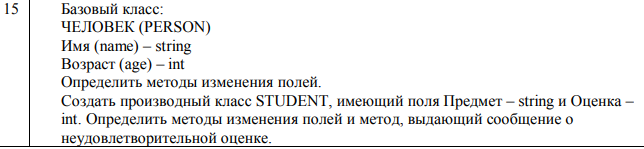
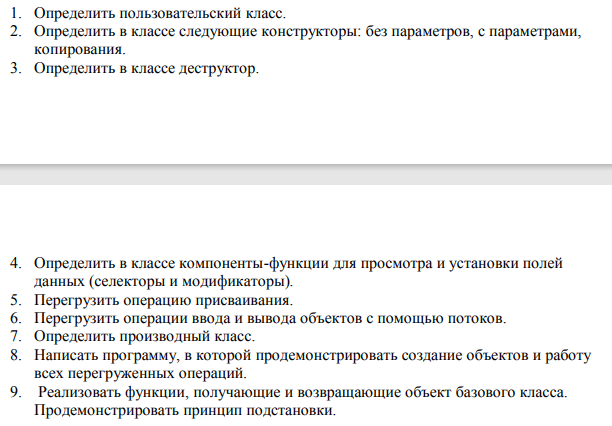
**Лабораторная работа**

**"Классы №4”**

Выполнил:   
студент группы РИС-23-1б   
Агзамов Артур Альферович

Проверила:   
доцент кафедры ИТАС   
О.А. Полякова

2024 г.

**Задача:**

**Анализ задачи:**

1. Создание производного класса Student, унаследованного от базового класса Person.

2. Определение дополнительных полей и методов для класса Student.

3. Реализация методов изменения полей и метода, выдающего сообщение о неудовлетворительной оценке.

4. Написание программы для демонстрации работы классов Person и Student, включая принцип подстановки объектов базового класса.

**Код:**

// int main()  
#include<iostream>

#include<string>

#include "class-4.h"

using namespace std;

int main(){

student chel("Vanya",18,"math",4);

cout<<chel;

chel.SetGrade(2);

chel.check\_grade();

cin>>chel;

cout<<chel;

return 0;

}

// class04.h  
#pragma once

#include<iostream>

#include<string>

using namespace std;

class person{

protected:

string name;

int age;

public:

person(string name, int age){

this->name= name;

this->age=age;

cout<<"Constructor person"<<endl<<endl;

}

person(const person& Person){

this-> name =Person.name;

this-> age= Person.age;

}

~person(){

cout<<"Constructor person"<<endl<<endl;

}

void SetName(string name){

this->name=name;

}

void SetAge(int age){

this->age=age;

}

string get\_name() {

return name;

}

int get\_age() {

return age;

}

};

class student:public person{

private:

string subject;

int grade;

friend ostream& operator <<(ostream& out, const student & Student);

friend istream& operator >>(istream& in, const student& Student);

public:

student(string name, int age, string subject, int grade) : person(name, age){

this->subject= subject;

this->grade=grade;

this->name=name;

this->age=age;

cout<<"Constructor student"<<endl<<endl;

}

student(const student& Student): person(name, age){

this->name=Student.name;

this->age=Student.age;

this->subject=Student.subject;

this->grade=Student.grade;

}

void operator = (const student& Student){

this->name=Student.name;

this->age=Student.age;

this->subject=Student.subject;

this->grade=Student.grade;

}

~student(){

cout<<"destructor student"<<endl<<endl;

}

void SetSubject(string subject){

this->subject=subject;

}

void SetGrade(int grade){

this->grade=grade;

}

string get\_subject(){

return subject;

}

int get\_grade(){

return grade;

}

void check\_grade(){

if (grade<3){

cout<<"student "<<name<<" has unsatisfactory grade in " << subject << endl;

}

else{

cout << "Student " << name << " has passed " << subject << " with grade " << grade << endl;

}

}

};

ostream& operator <<(ostream& out, const student & Student){

out << endl <<"Person:"<< Student.name<<" Age: "<<Student.age<<" Have grade "<<Student.grade<<" in subject "<<Student.subject<<endl;

out<<endl;

return out;

}

istream& operator >> (istream& in, student& Student){

cout<<"Enter person, his age, subject and his grade in subject"<<endl;

string name;

int age;

string subject;

int grade;

in>>name>>age>>subject>>grade;

Student.SetName(name);

Student.SetAge(age);

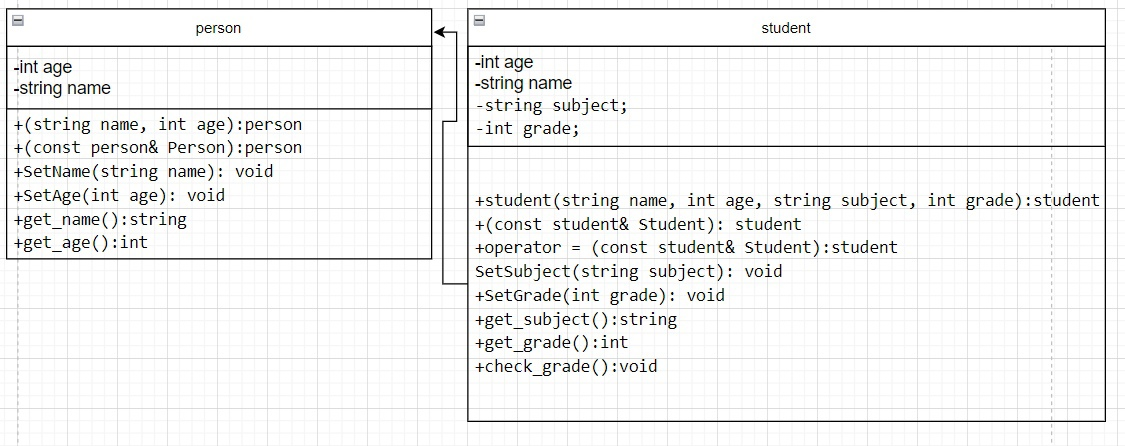
Student.SetSubject(subject);

Student.SetGrade(grade);

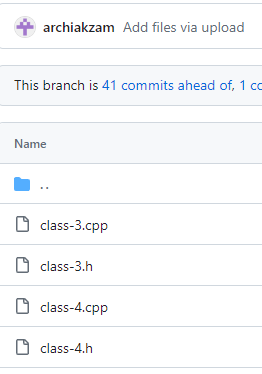
return in;

}

**UML диаграмма:**



Скрины из git:



**Выводы:** программа сработала корректно и вывела все возможные решения.